# PAIRWISE TEST SUITE GENERATOR TOOL BASED ON HARMONY SEARCH ALGORITHM (HS-PTSGT)

LAI YI XIANG

BACHELOR OF SCIENCE

(SOFTWARE ENGINEERING)

UNIVERSITI MALAYSIA PAHANG

## ABSTRACT

In the era of technology, software systems are around us in our daily life. Thus, a good system is needed to ensure the well function of gadget and devices with software embedded. However, exhaustive testing that required lot of time and financial resources is normally impossible. This thesis is about the research on developing a pairwise test suite generator tool based on Harmony Search Algorithm (HS-PTSGT) to generate optimum test suite. Test suite which considers all interaction of at most two factors will be generated from two-valued input parameters that are not more than 500. Users are allowed to enter parameter and values through the graphical user interface, and test cases will be generated through the execution of Harmony Search Algorithm involved random initialization of harmony memory and harmony memory improvisation. From the comparison with other existing pairwise testing tools in term of size, it is shown that the strategy used in HS-PTSGT can effectively generate optimum test suite with minimized test data and performed better on certain parameter size.

# ABSTRAK

Pada era teknologi ini, system perisian berada di sekeliling kita. Jadi, sistem perisian yang berkualiti amat diperlukan untuk memastikan peralatan yang terbenam perisian sentiasa berfungsi dengan baik. Zalimnya, ia adalah mustahil untuk kita menguji sesuatu sistem dengan sepenuhnya memandangkan masa dan wang yang banyak diperlukan. Tesis ini akan membincang tentang uji kaji untuk mencipta *pairwise test suite generator tool based on Harmony Search Algorithm (HS-PTSGT)* bagi menghasilkan koleksi set pengujian yang optimum. Koleksi set pengujian yang mempertimbangkan semua interaksi antara dua faktor tidak melebihi 500 parameter akan dihasilkan. Pengguna boleh masukkan parameter berserta dengan nilainya ke dalam *HS-PTSGT*, dan koleksi set pengujian akan dihasilkan setelah operasi *Harmony Search Algorithm* yang melibatkan *harmony memory initialization* dan *harmony memory improvisation*. Daripada keputusan kajian, *HS-PTSGT* terbukti mampu menghasilkan koleksi set pengujian yang optimum dengan berkesan dan berprestasi lebih baik daripada *pairwise testing tool* yang lain di keadaan yang tertentu.

# TABLE OF CONTENTS

# LIST OF TABLES

## LIST OF FIGURES

# LIST OF ABBREVIATIONS

HM : Harmony Memory

HMS : Harmony Memory Size

HMCR : Harmony Memory Considering Rate

PAR : Pitch Adjustment Rate

GA : Genetic Algorithm

ACA : Ant Colony Algorithm

IPO : In-Parameter-Order

RAD : Rapid Application Development

CASE : Computer-Aided Software Engineering tools

# CHAPTER 1

# INTRODUCTION

## 1.1 INTRODUCTION

In the era of technology, software systems are around us in our daily life. Electronic gadgets and devices that we're using are embedded with software, any bugs in these systems can cause the gadgets and devices to be malfunction. For critical system, the failure bring by software system may cause the loss of life and huge sum of money. Thus, a good system is needed to ensure the well function of these gadget and devices.

Start from the time when the term 'bug' is used to describe flaw in a system by the famous Thomas Alva Edison, defect in software system has become a threat in software development (Meerts, J. and Graham, D., 2013). In software development process, software testing plays an important role and required the most financial resources. Software Testing known as the process of discovers errors in system execution and validating the system against its specification (Myers, G. J., 1979).

During testing phase, test suite that with a collection of test cases are used to explore defects by entering test data as input into the system. In order to have effective testing on system, the generation of an optimum test suite is essential to avoid exhaustive testing which are practically impossible. This research is to develop a pairwise test suite generator tool based on Harmony Search Algorithm (HS-PTSGT).

Pairwise testing is a combinatorial technique used to reduce the number of test case inputs to a system, consider all interactions of at most two factors (McCaffrey, J. D., 2009). Pairwise testing normally begins by choosing individual inputs variables using domain partitioning; the values are then permuted to ensure the consideration of all pair interactions (Bach, J. and Schroeder, P. J, 2004). From the studies of Kuhn, D.R. and Reilly, M.J. (2002), more than 70% of bugs caused by two or fewer interaction condition able to be noticed.

Harmony search algorithm is a music-based metaheuristic optimization algorithm, divined from the aim to search for prefect state of harmony in jazz musician's improvisation process (Yang, X. S., 2009). Harmony Search algorithm works as follow: stochastic generation of an initial population harmony vectors and stored these vectors in a harmony memory. After that, new candidate harmony will be created based on all solutions in the harmony memory using a memory consideration rule, a pitch adjustment rule and a random re-initialization. Lastly, the new candidate harmony is compared with the worst harmony vector in the harmony memory. The new candidate harmony will replace the worst harmony and update the harmony memory if it is better than0 the worst harmony. This process will repeat until certain termination criterion is met. Harmony Search is a generalized optimization method for continuous, discrete, and constrained optimization and has been applied to numerous types of optimization problems. Further information will be discussed in details in Chapter 2.

## 1. 2 PROBLEM STATEMENT

Testing is required to find faults in a software product. For a big system, exhaustive testing that required lot of time and financial resources is impossible. Figure 1.1 shows the display tab in the options dialog for Microsoft Word. There are 17 possible options that can take two values which are check and uncheck. Hence, the combinations that need to be tested are $2^{17}$ or 131,072. If a test case need 5 minutes to be test, to completely test only the display tab will need approximately 15 months, which are practically inefficient.

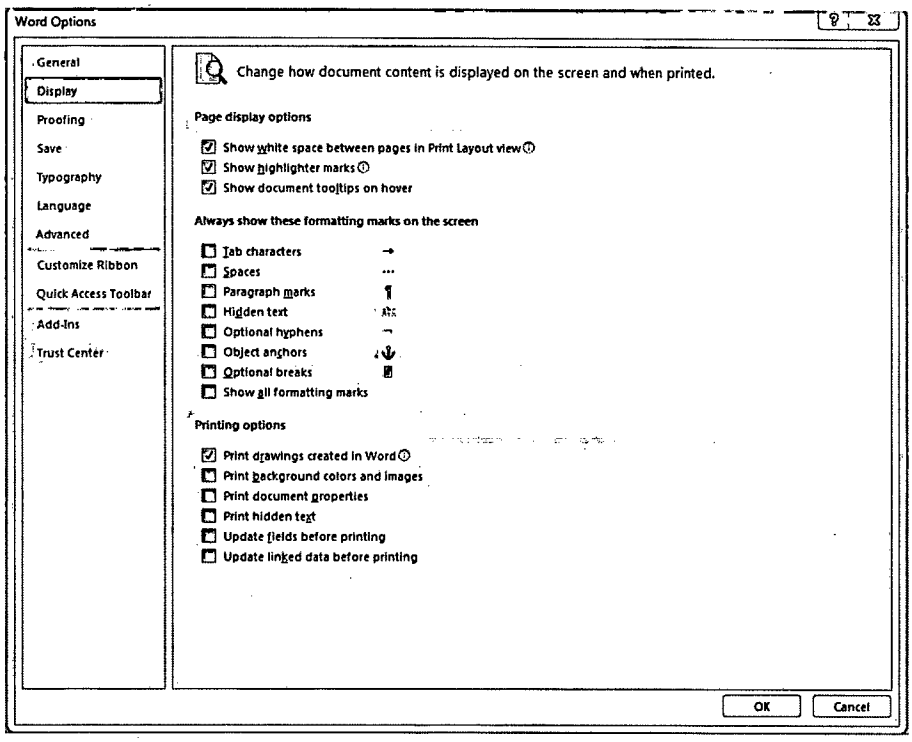


**Figure 1.1:** Display Tab in Options Dialog for Microsoft Word

## 1.3 GOAL AND OBJECTIVES

The goal for this research is to develop a pairwise test suite generator tool based on Harmony Search Algorithm. Several objectives that have been set to achieve the goal are as follow:

i) To apply Harmony Search Algorithm in generating pairwise test suite.
ii) To generate optimum test suite with minimized test data based on interactions of two factors.
iii) To evaluate the performance of this pairwise test suite generator tool in term of test size against other existing pairwise testing strategies.

## 1.4 RESEARCH SCOPE

In the process of development, NetBeans IDE 8.0 will be used. The scopes of this research are as follow:

i) Pairwise test suite generating based on Harmony Search Algorithm.
ii) Pairwise test suite generating with 2-valued input parameters that are not more than 500.
iii) Pairwise test suite generator tool developed in Java language.

## 1.5 METHODOLOGY

Software methodology is the activities used to control and ensure the achievement of stated objectives and finally reach the goal of this research. The methodology that going to use in this research are Rapid Application Development model, Figure 1.2 below shows the phases in the model:

- Phase 1: Requirements Planning
  - Literature review is performed in this phase to get the idea on the design of HS-PTSGT. The other tools is collected to be used in evaluate the performance of this pairwise test suite generator tool in term of test size against other existing pairwise testing strategies.
- Phase 2: User Design
  - The architecture of HS-PTSGT will be design to implement Harmony Search Algorithm. The interface design of the generator tool will be carrying out to ensure good usability.
- Phase 3: Construction
  - After the design of HS-PTSGT, this tool will be developed. The concept of pairwise testing and harmony search algorithm will be implemented in this tool, and test will be carry out to ensure correct implementation and well function of this tool.
- Phase 4: Cutover
  - The result of generated test suite of HS-PTSGT will be compared with other existing tools to show efficiency of this tool in term of test size. Study will also carry out to observe the effect of HMCR and PAR on test size. The whole research will then be documented. Future improvement will be suggested to improve this research.

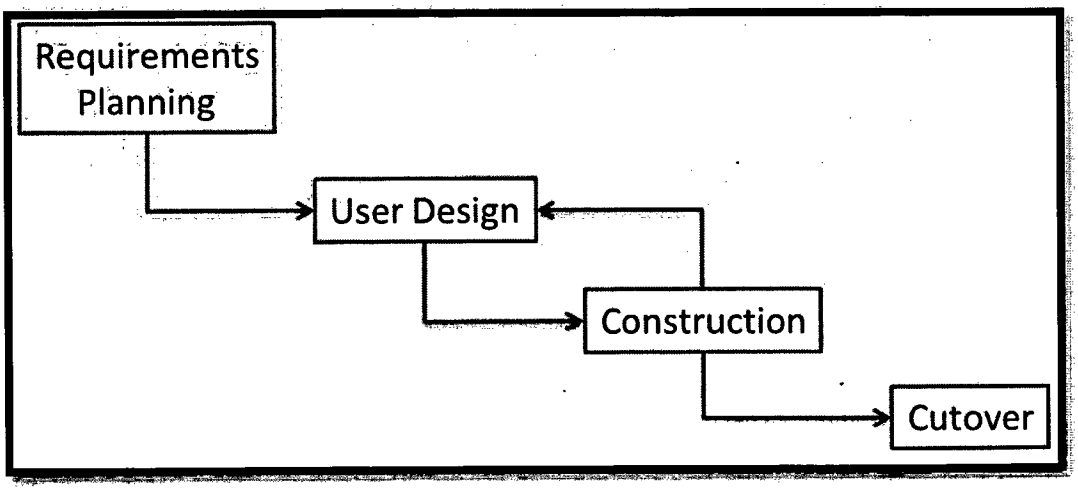


**Figure 1.2:** Rapid Application Development (RAD) Model

## 1. 6 THESIS ORGANIZATION

This thesis consists of seven (7) chapters, each chapter discuss about its own issues. Chapter 1 is Introduction. This chapter discusses on introduction about the research that will be done. The introduction, problem statement, goal, objectives, and scopes will be known.

Chapter 2 is Literature Review. Existing research and literature review that related to the project will be discussed in this chapter.

Chapter 3 is Methodology. In this chapter, overall approach and framework of research will be discussed. The method, technique or approach to be used for the project will be fixed.

Chapter 4 is Design of HS-PTSGT. This Chapter will illustrate the framework and model through flow work.

Chapter 5 is Implementation. All the processes involved in the research development will be recorded.

Chapter 6 is Result and Discussion. The results of pairwise test suite generator tool will be discussed on this chapter to determine the development is success or not. The test suite generated will be compared with other strategies in term of test size.

Chapter 7 is Conclusion. In this chapter, summarization of the research being done will be discussed. Besides that, the future suggestion and enhancement of the research topic also proposed in this chapter.

## 1.7 SUMMARY

This chapter has discussed the introduction of research on Pairwise Test Suite Generator Tool Based on Harmony Search Algorithm. Included in this chapter are problem statement, goal and objective, research scope, and thesis organization. The next chapter is literature review that discusses about the survey of existing pairwise testing strategy and Harmony Search Algorithm techniques.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 INTRODUCTION

In Chapter 1, the introduction of this research has been discussed, including problem statement, goal and objective, scope and methodology. Building on the material in Chapter 1, this chapter will focus on relevant literature review survey. In particular, pairwise testing techniques, Harmony Search Algorithm, and existing pairwise testing strategy are elaborated to justify the current work.

## 2.2 OVERVIEW

The main objective of this research is to generate optimum test suite with minimized test data based on interactions of two factors. To achieve this goal, pairwise testing and Harmony Search Algorithm will be implemented into this test suite generator tool.

### 2. 2. 1 Pairwise Testing

Pairwise testing is a combinatorial technique used to reduce the number of test case inputs to a system, consider all interactions of at most two factors (McCaffrey, J. D., 2009). In pairwise testing test suite that consider all combinations of the selected test data values for each pair of variables is generated. To understand the pairwise testing, following example is illustrated.

**Table 2.1:** Input Parameter and Value Illustration

| A | B | C | D |
|---|---|---|---|
| True | True | True | True |
| False | False | False | False |

Table 2.1 shows the input for a four plugs socket. In this system, there are 4 inputs available to user with 2 options respectively. Here, the exhaustive combination of all possible test data can be calculated by [number of test data = (number of value) $^{\text{number of input parameters}} = 2^4 = 16$]. The exhaustive combinations of this four plugs socket are shown in Table 2.2.

**Table 2.2:** Exhaustive Combination of 4 Inputs

| | Input Variables | | | |
|---|---|---|---|---|
| | **A** | **B** | **C** | **D** |
| Base Values | True | True | True | True |
| | False | False | False | False |
| Exhaustive Combination | True | True | True | True |
| | True | True | True | False |
| | True | True | False | True |
| | True | True | False | False |
| | True | False | True | True |
| | True | False | True | False |
| | True | False | False | True |
| | True | False | False | False |
| | False | True | True | True |
| | False | True | True | False |
| | False | True | False | True |
| | False | True | False | False |
| | False | False | True | True |
| | False | False | True | False |
| | False | False | False | True |
| | False | False | False | False |

In order to reduce the test data, pairwise testing technique can be used. In this technique, the 2-way possible combinations are AB, AC, AD, BC, BD, CD. For AB combination the total test size is reduced to 4 as only parameter A and B are considered while parameter C and D are treated as don't care value (see Table 2.3). Thus, C and D can randomly take either value True or False.

**Table 2.3:** 2-Way Combination for AB

| | Input Variables | | | |
|---|---|---|---|---|
| | **A** | **B** | **C** | **D** |
| Base Value | True | True | True | True |
| | False | False | False | False |
| 2-way combination for AB | True | True | True | True |
| | True | False | False | False |
| | False | True | True | True |
| | False | False | False | False |

The second combination is AC. For this combination the total test data also reduced to 4 since only parameter A and parameter C are considered. Parameter B and D can take either value of True or False randomly as they are treated as don't care values (see Table 2.4).

**Table 2.4:** 2-Way Combination for AC

| | Input Variables | | | |
|---|---|---|---|---|
| Base Value | **A** | **B** | **C** | **D** |
| | True | True | True | True |
| | False | False | False | False |
| 2-way combination for AC | True | True | True | True |
| | True | False | False | False |
| | False | True | True | True |
| | False | False | False | False |

The third combination is AD. For this combination the total test data also reduced to 4 since only parameter A and parameter D are considered. Parameter B and C can take either value of True or False randomly as they are treated as don't care values (see Table 2.5).

**Table 2.5:** 2-Way Combination for AD

| | Input Variables | | | |
|---|---|---|---|---|
| Base Value | **A** | **B** | **C** | **D** |
| | True | True | True | True |
| | False | False | False | False |
| 2-way combination for AD | True | True | True | True |
| | True | False | False | False |
| | False | True | True | True |
| | False | False | False | False |

The fourth combination is BC, the total test data also reduced to 4 since only parameter B and parameter C are considered. Parameter A and D can take either value of True or False randomly as they are treated as don't care values (see Table 2.6).

**Table 2.6:** 2-Way Combination for BC

| Base Value | Input Variables | | | |
|---|---|---|---|---|
| | A | B | C | D |
| | True | True | True | True |
| | False | False | False | False |
| 2-way combination for BC | True | True | True | True |
| | True | True | False | False |
| | False | False | True | True |
| | False | False | False | False |

In the fifth combination (BD), the total test data again reduced to 4 since only parameter B and parameter D are considered. Parameter A and C can take either value of True or False randomly as they are treated as don't care values (see Table 2.7).

**Table 2.7:** 2-Way Combination for BD

| Base Value | Input Variables | | | |
|---|---|---|---|---|
| | A | B | C | D |
| | True | True | True | True |
| | False | False | False | False |
| 2-way combination for BD | True | True | True | True |
| | True | True | False | False |
| | False | False | True | True |
| | False | False | False | False |

Similar to previous combinations, the total test data for CD combination is reduced to 4. This is because only C and D are considered. A and B are then treated s don't care value where they can randomly take either True or False values (see Table 2.8).